Planning the Workday of Bus Drivers by a Graph List-Coloring Model

M. Lucci[1], G. Nasini[1,2], and D. Severín[1,2]

[1] FCEIA, Universidad Nacional de Rosario, Argentina.
[2] CONICET, Argentina.
maurolucci.14@gmail.com
{nasini,daniel}@fceia.unr.edu.ar

Abstract. In this work, we address the problem of planning the workday of bus drivers in argentinian intercity bus transport companies. In particular, we focus on a company which needs to fulfill roughly 800 trips per day between 3 cities of the Province of Buenos Aires with a stuff of around 200 drivers and 100 buses. Planning consists of assigning one driver to each trip in a way the driver performs all the trips without scheduling conflicts and minimizing the overall amount of overtime among all bus drivers.
We model the problem as a particular Graph Coloring Problem and we propose an Integer Linear Programming formulation. Computations experiments show that this formulation outperforms other ones given in the literature for the same problem.
In order to address large instances as the one given by the company, we also propose a heuristic algorithm that delivers better solutions than the company actually uses in a reasonably amount of time. The heuristic has two phases where the first one constructs an initial solution and the second one improves the solution iteratively.

Keywords: Integer programming model · Graph coloring · Heuristics · Planning workday of bus drivers

1 Introducción

Entre las aplicaciones de técnicas de Investigación Operativa, las asociadas a problemas relativos a la planificación del transporte están entre las de mayor impacto económico [1]. En este trabajo en particular, abordamos el problema de diseño de jornadas laborales de conductores en empresas de transporte público.

Estas empresas tienen adjudicados un conjunto de viajes que deben cumplir diariamente, definidos por lugar de origen, lugar de destino, horario de salida y duración. Los convenios laborales de los conductores establecen restricciones respecto a cuáles y cuántos viajes puede realizar un conductor en su jornada. Entre ellas podemos mencionar las duraciones máximas de jornadas, los descansos obligatorios entre viajes y las diferencias entre horarios diurnos y nocturnos. Las empresas también imponen sus políticas como duraciones máximas de los descansos y duraciones mínimas de las jornadas.

En la mayoría de los países, los elevados costos operativos implicados en el transporte público son parcialmente subsidiados por el Estado. Dichos subsidios en general son insuficientes, motivando la necesidad de reducir los costo sin perjudicar la calidad en el servicio. En base a esto, el problema de diseñar eficientemente las jornadas laborales de los conductores se convierte en la clave para el funcionamiento de estas empresas [2].

El diseño de jornadas laborales de conductores puede enmarcarse entre los problemas de Planificación de Tareas, en inglés *Duty Scheduling* (DS). El DS ha sido estudiado ampliamente a lo largo de los últimos 40 años. En A. Ernst et al. [3] se hace una extensa enumeración de diversas publicaciones que lo abordan desde distintos enfoques.

En M. Grötschel et al. [4] se discuten los diversos algoritmos que conforman el *DS-OPT*, una herramienta de optimización basada en generación de columnas que forma parte de uno de los productos comerciales vigentes en el mercado para la planificación en el transporte [5].

Si bien los productos comerciales en general permiten al usuario cierto nivel de personalización sobre los criterios de la optimización, cada país o empresa posee sus propias normativas, las cuales pueden llegar a ser muy diversas, y con frecuencia no permiten modelarlas en su totalidad, generando así que las soluciones otorgadas no siempre sean óptimas, o incluso no puedan ser implementadas en la práctica.

Tal es el caso de una empresa de transporte interurbano de la Provincia de Buenos Aires sobre la cual estudiamos el problema de diseño de jornadas laborales. Este ya fue abordado en dos trabajos anteriores (véase [6, 7]) y, en esta ocasión, presentamos un nuevo modelo que se desempeña significativamente mejor que los propuestos en los trabajos mencionados.

A continuación presentamos notaciones y definiciones que serán necesarias para la mejor comprensión del trabajo.

Denominamos con V al conjunto de viajes adjudicados a la empresa y llamamos A al conjunto de pares de viajes u y v tales que un mismo conductor puede tomar el viaje v inmediatamente después de haber realizado u. En principio, para que esto sea posible, la hora de salida de v debe ser posterior a la hora de llegada de u. Más aún, debe haber suficiente tiempo para que el conductor realice el descanso establecido y para que deje el coche en el punto de origen de v en caso de diferir del destino de u.

Llamamos *jornada laboral* a todo subconjunto de viajes que pueden ser asignados a un mismo conductor en un mismo día, respetando todos los convenios laborales y la política de la empresa. En particular, existen restricciones respecto a la duración mínima y máxima de una jornada y el lugar de origen y de finalización deben coincidir. Ambas condiciones quedan determinadas por los datos de sus primer y último viaje. De esta manera, construimos un conjunto J con aquellos pares de viajes que pueden ser primer-último viaje de una jornada laboral.

Como el objetivo es minimizar las horas laborales, el costo de cada jornada está asociada a su duración y sólo depende de sus viajes inicio y fin. Así, para

cada $(a, b) \in J$, su costo c_{ab} se compone de un costo fijo (asociado al salario mensual) y el costo de las horas extras requeridas (si las hubiera).

En este sentido, a cada jornada laboral $j \subset V$ con primer-último viaje $(a, b) \in J$, nos referimos a ella como ab-jornada, le asignamos un costo $c(j) = c_{ab}$. Por último, llamamos J-jornada a cualquier ab-jornada con $(a, b) \in J$. Así, cada posible diseño de jornadas laborales corresponde a una partición del conjunto de viajes V en J-jornadas. El costo de cada partición es la suma de los costos de las jornadas involucradas.

Bajo este contexto, definimos formalmente el *Problema de Diseño de Jornadas Laborales de Conductores* (DJLC):

PROBLEMA DJLC
INSTANCIA: $V, A, J \subset V \times V$, $c \in \mathbb{R}_+^J$.
OBJETIVO: Determinar una partición de V en J-jornadas, de costo mínimo.

Resulta natural la siguiente transformación de DJLC a un problema de optimización en digrafos.

Dada una instancia (V, A, J, c) de DJLC, construimos el digrafo $D = (V, A)$. Dado $(a, b) \in J$, los vértices de todo ab-camino (dirigido) en D se corresponden a viajes que se pueden realizar de manera consecutiva por un mismo conductor, determinando así una posible jornada laboral, y viceversa. Llamando J-camino a cualquier ab-camino en D con $(a, b) \in J$, existe una correspondencia $1-1$ entre J-jornadas y J-caminos.

Definiendo c_{ab} el costo de cada ab-camino en D, con $(a, b) \in J$, nuestro problema puede ser reformulado de la siguiente manera:

PROBLEMA DJLC
INSTANCIA: $D = (V, A)$ digrafo, $J \subset V \times V$, $c \in \mathbb{R}_+^J$.
OBJETIVO: Determinar una partición de V en J-caminos, de costo mínimo.

Una restriccion particular que se impone la empresa es que los viajes involucrados en una misma jornada laboral comiencen todos el mismo día. La misma justifica esta decisión en, por un lado, el alto costo que tendría una jornada que se extienda desde la noche hasta la madrugada, pues los minutos de conducción nocturnos son más costosos que los diurnos. Por otro lado, la empresa cambia las frecuencias de sus viajes los fines de semana, haciendo que una planificación diaria sea más adecuada para efectuar de manera ordenada esta transición.

Así, al ordenar los viajes de V en forma creciente según su hora de salida, se satisface que todo arco de A va de un viaje anterior a uno posterior en aquel orden. En consecuencia, el digrafo D es acíclico y además la hora de salida de los viajes determina un orden para sus nodos con la propiedad mencionada recientemente (conocido en la literatura como *orden topológico*).

La complejidad computacional de DJLC no se conoce. Sin embargo, si relajamos la condición de aciclicidad en el digrafo de entrada, el problema resulta $\mathcal{NP}$-difícil ya que el *Problema de Existencia de Camino Hamiltoniano Dirigido* (CHD) puede reducirse polinomialmente a su problema de decisión asociado.

En efecto, dado un digrafo $D = (V, A)$, donde nos preguntamos si existe un camino hamiltoniano dirigido, construimos una instancia del problema de

decisión asociado de la siguiente manera: el digrafo es el mismo D, $J = \{(u, v) \in V \times V : u \neq v\}$ y $c_{uv} = 1$ para todo $(u, v) \in J$.

Es fácil ver que D tiene un camino hamiltoniano si y sólo si existe una partición de V en J-caminos cuyo costo total es a lo sumo 1.

Como el problema CHD es $\mathcal{NP}$-completo [8], DJLC resulta $\mathcal{NP}$-difícil.

En [6], se propuso una formulación del DJLC como un problema de *Programación Lineal Entera* (PLE) a partir de variables de asignación conductor-viaje. Nos referimos a la misma como M_0. Utilizando CPLEX [9] como solver de PLE, en el término de 2 horas sólo era posible resolver instancias de hasta 40 viajes, mientras que las instancias más pequeñas de la empresa tenían cerca de 500 viajes. Sin embargo, su eficiencia a la hora de resolver instancias pequeñas hizo posible incorporar este modelo en el diseño de una heurística de mejora que, aplicada sobre las soluciones que implementaba la empresa, permitió obtener soluciones que mejoraban las mismas.

Luego, en [7], se presentaron dos nuevas formulaciones de PLE (que se basan en la idea de particionar un digrafo en circuitos dirigidos). Nos referimos a ellas como M_1 y M_2. La formulación M_1, al igual que M_0, es compacta. Es decir, tiene tamaño polinomial respecto a la entrada, utilizando $O(|V|^2 \times k)$ variables, donde k es una cota superior de la cantidad de conductores y forma parte de la entrada. Por otro lado, M_2 utiliza sólo $O(|V|^2)$ variables pero requiere un número exponencial de restricciones, las cuales deben ser gestionadas como cortes durante la optimización. Ambas formulaciones logran resolver instancias del cuádruple de tamaño que M_0 (hasta 160 viajes) pero M_2 resultó ser mucho más rápida que M_1. Reemplazando M_0 por M_2 en la heurística de mejora antes mencionada, las mismas soluciones "mejoradas" pueden ser obtenidas en tiempos significativamente menores.

En este trabajo, proponemos un modelo del DJLC como Problema de Coloreo por Listas en Grafos. Desarrollamos una formulación de PLE que llamamos M_3, también compacta, y que resuelve por optimalidad instancias de hasta 280 viajes, en dos horas. También pudimos resolver por optimalidad una de las instancias de la empresa correspondiente a 587 viajes, en 45 horas. Además, con el objetivo de abordar instancias mayores, diseñamos una heurística de dos fases: la primera construye una solución para la empresa donde todas sus jornadas cumplen con las restricciones impuestas y la segunda mejora la solución existente de forma iterativa.

2 Coloreo por Listas y DJLC

Un coloreo f en un grafo $G = (V, E)$ es una función que a cada vértice de V le asigna un color de forma tal que a vértices adyacentes le corresponden colores distintos. En un *coloreo por listas* de G [10], el color asignado a cada vértice v no puede ser cualquiera sino que debe pertenecer a una lista predeterminada L_v de colores permitidos.

Así, dados $G = (V, E)$ y $L = \{L_v : v \in V\}$, un *L-coloreo por listas de G* es un coloreo f de G tal que $f(v) \in L_v$ para todo $v \in V$.

Sea $\mathcal{C} = \cup_{v \in V} L_v$ el conjunto de colores. Para todo $j \in \mathcal{C}$ definimos $V_j = \{v \in V : j \in L_v\}$ y G_j como el subgrafo de G inducido por V_j. Un conjunto $S \subset V$ es un estable de G si para todo $u, v \in S$ resulta $uv \notin E$. Claramente, en un L-coloreo por lista, los vértices pintados con j forman un estable de G_j para todo $j \in \mathcal{C}$.

Si a cada color $j \in \mathcal{C}$ le asignamos un costo, el problema de optimización asociado consiste en encontrar un L-coloreo por listas de G que minimice la suma de los costos de los colores utilizados.

En DJLC, cada viaje de V debe ser asignado a una posible jornada laboral. Así, podemos pensar esta asignación como un coloreo de un grafo $G = (V, E)$ donde los vértices son los viajes y el conjunto de colores disponibles $\mathcal{C}$ es el conjunto de pares en J. Sin embargo, dado un viaje v, no todos los pares de J corresponden a jornadas a las que v puede pertenecer. Por ejemplo si la hora de finalización de v es posterior al inicio del viaje b, claramente v no puede pertenecer a una ab-jornada. Surge pensar entonces en un coloreo por listas donde cada vértice (viaje) tiene asociado una lista de colores (jornadas) permitidos.

Resulta oportuno mencionar que este último ejemplo está bien fundamentado debido a que el digrafo D, mencionado en la introducción, es acíclico y la hora de salida asociada a cada uno de sus nodos define un orden topológico. El digrafo acíclico de la figura 1, donde sus nodos se ubican de izquierda a derecha según la hora de salida del viaje que representan, no cumple con la condición referida al orden topológico y está en contradicción con el ejemplo enunciado anteriormente.

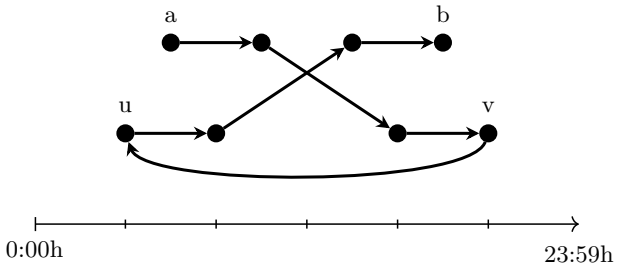

Fig. 1.

Dado $v \in V$, definimos L_v como el conjunto de los pares $(a, b) \in J$ tales que es posible construir una ab-jornada que incluya a v. En términos del digrafo D, esto es equivalente a que exista un ab-camino que contenga a v, o bien, que existan un av-camino y un vb-camino en D. Esta última equivalencia es correcta debido a que las propiedades estructurales que posee D hacen que la concatenación de dichos caminos genere un ab-camino simple.

El conjunto de arcos E de G se corresponde con aquellos pares de viajes que no pueden pertenecer simultáneamente a una misma jornada laboral. Para que dos viajes u, v puedan pertenecer a la misma jornada $(a, b) \in J$ necesitamos que

$(a, b) \in L_u \cap L_v$ y además que exista un camino entre u y v en D. Por lo tanto, $(u, v) \in E$ si $L_u \cap L_v \neq \emptyset$ y no existe en D un uv-camino ni un vu-camino.

Con estas definiciones, es fácil probar el siguiente resultado:

Proposition 1. *Toda partición de V en J-jornadas determina un L-coloreo por listas de $G = (V, E)$.*

Proof. Dada una partición de V en J-jornadas, cada $v \in V$ pertenece a una única ab-jornada de dicha partición. Claramente $(a, b) \in L_v$ y asignamos a v el color (a, b).

Para probar que esta asignación determina un coloreo, debemos probar que si $uv \in E$, u y v no están en la misma jornada de la partición. Claramente, si u y v están en una misma jornada, debe existir un camino en D que los una. Por definición de E tal camino no puede existir con lo cual u y v tendrán colores distintos.

Describimos una formulación de PLE para el problema de Coloreo por Listas en G, basada en una formulación para el Coloreo Clásico que ha resultado muy competitiva [11].

Las variables de asignación en consideración son, dado un vértice $v \in V$ y un color $j \in L_v$, la variable $x_{vj} = 1$ si y solo si v se pinta con j; dado un color $j \in \mathcal{C}$, la variable $w_j = 1$ si y solo si algún vértice es pintado con j.

El objetivo resulta entonces minimizar

$$\sum_{j \in J} c_j w_j$$

sujeto a las siguientes restricciones.

- *Restricciones de asignación.* Cada $v \in V$ se debe pintar con un único color de L_v:
$$\sum_{j \in L_v} x_{vj} = 1. \tag{1}$$
- *Restricciones de arco.* Para cada $(u, v) \in E$ y $j \in L_u \cap L_v$, es posible pintar con j a lo sumo uno de ellos:
$$x_{uj} + x_{vj} \leq w_j. \tag{2}$$
- *Restricciones sobre vértices aislados.* Para cada $v \in V$ y $j \in L_v$ tal que v es un vértice aislado de G_j, es necesario agregar la siguiente restricción para controlar la activación de w_j:
$$x_{vj} \leq w_j. \tag{3}$$
- *Condiciones de integralidad.* Para todo $v \in V$ y $j \in L_v$,
$$x_{vj} \in \{0, 1\}, \tag{4}$$
y para todo $j \in \mathcal{C}$,
$$w_j \in \{0, 1\}. \tag{5}$$

Observemos que si $w_j = 1$, entonces debe existir al menos un vértice coloreado con j. Esto se puede garantizar con las restricciones

$$\sum_{v:j\in L_v} x_{vj} \geq w_j$$

para todo $j \in \mathcal{C}$. Sin embargo, esto se da naturalmente si la solución es óptima y, por otra parte, incorporar este tipo de restricciones no produce ninguna mejora apreciable, por lo que no las tendremos en cuenta.

2.1 Restricciones Adicionales

Para tener una correspondencia $1-1$ entre las soluciones del PLE correspondiente a las restricciones (1)-(5) y las particiones de V en J-jornadas deberemos considerar algunas restricciones adicionales. Esto se debe a que no todo L-coloreo por listas de G determina una partición de V en J-jornadas. Sin embargo podemos probar que todo L-coloreo determina un cubrimiento de V en J-jornadas en el siguiente sentido:

Proposition 2. *Dado un L-coloreo f de $G = (V, E)$, $(a, b) \in J$ y $C_{ab} = \{v \in V : f(v) = (a, b)\}$. Si $C_{ab} \neq \emptyset$ entonces existe una ab-jornada j tal que $C_{ab} \subset j$.*

Proof. Para todo $w \in C_{ab}$, como $f(w) = (a, b)$ existe un aw-camino y un wb-camino en D. Como D es acíclico, w es el único vértice en común de ambos caminos. Por lo tanto, existe al menos un ab-camino en D.

Debemos probar que existe una ab-jornada o, equivalentemente, un ab-camino P en D tal que los vértices de P, $V(P)$, incluyan a C_{ab}. Sea entonces P un ab-camino que maximiza $|V(P) \cap C_{ab}|$. Probaremos que $C_{ab} \subset V(P)$.

Supongamos entonces que existe $v \in C_{ab} \setminus V(P)$. Para todo $w \in C_{ab} \setminus \{v\}$, existe un wv-camino en D o un vw-camino en D. Particionamos los vértices de $V(P) \cap C_{ab}$ en los siguientes conjuntos:

$$V_1 = \{w \in V(P) \cap C_{ab} : \text{existe un } wv\text{-camino en } D\} \cup \{a\}$$

y

$$V_2 = \{w \in V(P) \cap C_{ab} : \text{existe un } vw\text{-camino en } D\} \cup \{b\}.$$

Como D es acíclico, $V_1 \cap V_2 = \emptyset$. Sea w_1 el último vértice en P que pertenece a V_1 y w_2 el primer vértice en P que pertenece a V_2, y llamamos P' al camino obtenido reemplazando el w_1w_2-camino en P por la concatenación del w_1v-camino y el vw_2-camino en D. Luego, $|V(P') \cap C_{ab}| > |V(P) \cap C_{ab}|$, resultando una contradicción.

Para que las soluciones factibles representen particiones (en jornadas) en vez de cubrimientos, incorporamos a la formulación las siguientes restricciones.

- *Restricciones de primer-último viaje de jornada.* Para $j = (a, b) \in J$, si el color j se usa, a y b deben estar pintados con j:

$$x_{aj} = x_{bj} = w_j. \tag{6}$$

– *Restricciones de camino.* Para todo $j = (a, b) \in J$ y $v \in V \setminus \{b\}$ tal que $j \in L_v$, si v se pinta con j alguno de sus vecinos de salida también debe estar pintado con j:

$$\sum_{u \in N_D^+(v)} x_{uj} \geq x_{vj}, \tag{7}$$

donde $N_D^+(v)$ denota el conjunto de vecinos de salida de v en D.

Llamamos M_3 a la formulación correspondiente a las restricciones (1)-(7).

3 Experimentos Computacionales

Las implementaciones necesarias fueron desarrolladas en lenguaje C++ utilizando CPLEX 12.5.1 [9] como herramienta principal para resolver los problemas de PLE. Los experimentos computacionales fueron llevados a cabo en una computadora utilizando sólo un thread de un procesador AMD Athlon II P340 a 2.2 Ghz, con 3 GB de memoria y sistema operativo Ubuntu 14.04 LTS.

3.1 Comparación con Modelos Previos

Comparamos la eficiencia de M_3 con M_2 sobre una instancia de 818 viajes, provista por la empresa de transporte interurbano ya mencionada, y sobre instancias generadas a partir de la misma.

La empresa considera la posibilidad de que un conductor pueda no realizar el descanso reglamentario después de un viaje (de corta duración) u, realizar a continuación un viaje v y realizar un descanso más prolongado al finalizar v (véase [6]). Para modelar esta situación en M_2, fue necesario incorporar un conjunto de viajes especiales, denominados *mellizos*. En efecto, el conjunto de viajes V se particiona en tres conjuntos V_1, V_2 y V_2', y existe una biyección entre V_2 y V_2' de forma tal que a cada $v \in V_2$ le corresponde un mellizo $v' \in V_2'$ y viceversa. Estos mellizos tienen la particularidad de que si $v \in V_2$ es asignado a una jornada laboral entonces v' no debe asignarse, y viceversa.

Así, es fácil adaptar la formulación M_3 para contemplar los mellizos. Las restricciones de asignación (1) deben ser reemplazadas por las siguientes.

$$\sum_{j \in L_v} x_{vj} = 1, \quad \forall v \in V_1$$

$$\sum_{j \in L_v} x_{vj} + \sum_{j \in L_{v'}} x_{v'j} = 1, \quad \forall v \in V_2$$

Por otro lado, la empresa no está dispuesta a aumentar el número de conductores de su planta. De esta manera, M_2 incluía originalmente una restricción que fijaba el número de jornadas laborales. En M_3 decidimos no incorporar esta restricción que parece dificultar la resolución de las relajaciones y, en su lugar, manejar el número de jornadas a través del costo fijo, de manera que en el óptimo

la cantidad de jornadas tienda a ser mínima. Para realizar las comparaciones, adaptamos la formulación de M_2 de manera tal de coincidir con este criterio.

Además, en M_3, optamos por que las restricciones (7) no formen parte la relajación inicial sino que son administradas como *lazy constraints*. Esto se debe a que rara vez son violadas por las soluciones enteras durante la optimización.

A continuación mostramos la diferencia en el mayor tamaño de instancias que son capaces de resolver por optimalidad las formulaciones M_2 y M_3, con un tiempo límite establecido en 2 horas. La metodología aplicada para generar las instancias, que llamaremos *subinstancias*, consiste en seleccionar de forma aleatoria una cantidad N de posibles J-jornadas disjuntas de la instancia real. La subinstancia quedará determinada por los viajes correspondientes a dichas jornadas. Generamos tres grupos, cada uno de ellos formado por diez subinstancias construidas de la forma previamente descripta, tomando $N = 10, 15, 20$ respectivamente, y las resolvemos mediante un esquema de *branch-and-bound (B&B)* puro, utilizando CPLEX con los cortes y heurísticas desactivados.

Los resultados se registran en la tabla 1. La primer columna contiene la cantidad de viajes de la subinstancia. Las columnas encabezadas con *Nodos o Gap* reportan la cantidad de nodos procesados en el B&B, en caso de que la instancia se haya resuelto antes del tiempo límite, o el gap relativo en caso contrario. Así mismo, usamos una marca "−" si durante la optimización no se logra encontrar una solución factible. Por último, en las columnas *Tiempo* denotamos la cantidad de segundos que requirió la optimización de la subinstancia. Podemos observar que tanto M_2 como M_3 resolvieron de inmediato todas las subinstancias del primer grupo. En el segundo grupo, mientras que M_2 resolvió todas las subinstancias en un tiempo promedio de 127 seg., M_3 las resolvió en menos de 1 seg. Respecto al último grupo, M_2 sólo resolvió una de ellas, a diferencia de M_3 que nuevamente resolvió cada una de ellas rápidamente. Es importante destacar que, para la mayoría de las subinstancias, M_3 no procesó ningún nodo. Esto significa que resolvió el problema en el nodo raíz sin necesidad de hacer ramificaciones. A partir de estos resultados, queda en evidencia la superioridad de M_3 respecto a M_2, al menos para este conjunto de subinstancias.

3.2 Nuevos Límites para Instancias de DJLC

Ahora nos proponemos analizar el mayor tamaño de subinstancia que la formulación M_3 resuelve por optimalidad en 2 horas. Para lograr un mayor rendimiento, utilizamos una formulación más ajustada reemplazando las restricciones de arco (2) por las denominadas *restricciones cliques*, habituales en las formulaciones de coloreo. Para cada color $j \in \mathcal{C}$, sea $\mathcal{Q}_j$ un cubrimiento de aristas de G_j por cliques maximales. Reemplazamos las restricciones de arco por:

Restricciones cliques: para toda clique $Q \in \mathcal{Q}_j$ y $j \in \mathcal{C}$, a lo sumo un vértice de Q puede ser pintado con j:

$$\sum_{v \in Q} x_{vj} \leq w_j.$$

De esta manera, obtenemos una formulación con menos cantidad de restricciones y más ajustadas. Por ejemplo, en el caso de la instancia real, la cantidad de restricciones de arco son alrededor de 4.6 millones mientras que las restricciones cliques están en el orden de 500 mil. Además, el valor objetivo de la relajación lineal se incrementa en un 1,75% al utilizar restricciones cliques.

Sería deseable conocer el conjunto $\mathcal{Q}_j$ de menor cardinal, pero es sabido que obtenerlo es $\mathcal{NP}$-difícil por lo que su generación se realiza con un procedimiento goloso.

Generamos tres grupos de diez subinstancias, tomando $N = 50, 60, 70$ respectivamente y las resolvemos con la siguiente combinación de parámetros de CPLEX: tanto las heurísticas como los cortes quedan activados y las relajaciones lineales se resuelven utilizando el método Barrier. Este algoritmo resuelve la relajación lineal de la instancia real en 290 segundos, mientras que los algoritmos del tipo Simplex requieren tiempos superiores a las 9 horas.

Reportamos los resultados obtenidos en la tabla 2 cuyo formato es similar a la tabla 1. Podemos observar que, aunque se presentaron dificultades para resolver el tercer grupo, se lograron encontrar buenas soluciones factibles en éste ya que el gap fue inferior al 1%. Concluimos que, con el nuevo modelo, se logran resolver instancias de hasta 280 viajes aproximadamente.

Motivados por este buen resultado referido al notable incremento del tamaño de subinstancias que M_3 resuelve en dos horas, probamos resolver la instancia de la empresa con 818 viajes. En un término de 48 horas no se logró encontrar siquiera una solución factible. Sin embargo, esto motivó la búsqueda de metodologías que nos permitieran mejorar los resultados para esta instancia, utilizando el potencial de M_3.

3.3 Mejores soluciones para la Empresa

En lo que sigue, llamaremos jornadas *legales* a aquellas que cumplen todas las restricciones y jornadas *fuera de convenio* a aquellas que no cumplen con alguno de los convenios o políticas de la empresa. Siempre es posible calcular el costo de una jornada fuera de convenio, aunque lo deseable es que no exista ninguna en la solución.

Para garantizar los 818 viajes adjudicados, la empresa posee una planta de 191 conductores. Con su actual metodología, no fue capaz de diseñar las 191 jornadas laborales de manera tal que todas cumplan los requerimientos señalados. Actualmente utiliza una solución con 126 jornadas legales y 65 fuera de convenio. Esta solución le obliga a pagar a la empresa 3956 minutos extras. A través de una heurística de mejora 3-opt basada en la resolución por optimalidad de pequeñas instancias con M_0 como formulación de PLE [6], se consiguió mejorar dicha solución, reduciendo la cantidad de jornadas fuera de convenio a 55 (y aumentando a 136 las jornadas legales) y los minutos extras a 3815. Este resultado representa una mejora del 15,4% respecto a cantidad de jornadas fuera de convenio y de 3,6% respecto a minutos extras. Se reportó un tiempo de ejecución para la heurística de 5256 segundos. Posteriormente, utilizando la formulación M_1 en

la resolución de pequeñas instancias, el tiempo de ejecución se pudo reducir a 2050 segundos [7].

Por lo evidenciado previamente, M_3 permitió aumentar ampliamente el mayor tamaño de subinstancia resuelta. En base a este resultado, generamos una subinstancia a partir de los 587 viajes de las 136 jornadas legales de la mejor solución obtenida hasta el momento. Claramente, al basarnos sólo en jornadas legales, tenemos garantía de que esta subinstancia es factible y pudo ser resuelta por optimalidad en un término de 45 horas. Al reemplazar en la mejor solución obtenida hasta el momento las 136 jornadas legales originales por las encontradas por optimalidad, se obtuvo una solución que requiere 3436 minutos extras, mejorando en un 13.1% los valores reportados por la empresa.

Claramente, esta última solución es óptima si no consideramos necesario reducir el número de jornadas fuera de convenio. Sin embargo, en virtud de que la heurística de mejora 3-opt había reducido este y también el número de horas extras, nuestro siguiente objetivo entonces fue el de construir soluciones con menor número de jornadas fuera de convenio. Para ello, desarrollamos una nueva heurística de mejora basada en el potencial de M_3.

Nueva heurística de mejora. La heurística de mejora parte de una solución inicial del problema. Así, dada la instancia y un conjunto de jornadas como solución inicial S (el cual puede tener jornadas fuera de convenio) se resuelven por optimalidad, utilizando M_3, n subinstancias correspondientes a los viajes de una partición $\{S_1, \ldots, S_n\}$ de S. Si la i-ésima subinstancia es factible, consideramos que su solución óptima S'_i mejora a S_i si utiliza menos minutos extras y tiene a lo sumo la misma cantidad de jornadas que S_i. En ese caso, construimos la solución S' correspondiente a reemplazar en S a S_i por S'_i.

La forma de construir dicha partición tiene un fuerte impacto en el despeño de la heurística. Originalmente generamos la misma en forma aleatoria, obteniendo subinstancias de rápida resolución pero con pequeños márgenes de mejora.

Un segundo criterio fue el agrupamiento de *jornadas cercanas*, medida la cercanía entre ellas por la cantidad de adyacencias de sus viajes en el digrafo D. Esta estrategia tiende a generar subinstancias con mayores márgenes de mejora, a cambio de mayores tiempos en su resolución.

En nuestro caso, generamos la n-partición mediante un esquema por rondas, combinando una estrategia aleatoria con una de cercanía. En efecto, en la primer ronda se reparte en cada S_i una jornada aleatoria de S. En las próximas rondas, en cada S_i se reparte la jornada de S que sea mas cercana a las jornadas previamente incluidas, es decir, cuyos viajes tengan más cantidad de adyacencias con los viajes correspondientes a las jornadas de S_i.

Comparamos el comportamiento de la nueva heurística de mejora con el 3-opt mencionado anteriormente. Para ello, tomamos como solución inicial la que utiliza actualmente la empresa, con 65 jornadas fuera de convenio y 3956 min extras. Repitiendo el procedimiento anterior 100 veces con $n = 50, 40$ conseguimos una solución con 51 jornadas fuera de convenio y 3776 minutos extras, en un tiempo de ejecución de 330 segundos. Respecto a los valores reportados por la

empresa, representa una mejora del 21,5% respecto a jornadas fuera de convenio y del 4,5% respecto a minutos extras. También representa una mejora del 7,3% respecto a jornadas fuera de convenio y del 1,0% respecto a minutos extras al compararla con la solución obtenida con 3-opt.

Heurística constructiva. Los enfoques anteriores no permitieron responder a la pregunta de si existirá una solución (no necesariamente óptima) sin jornadas fuera de convenio. Motivados por ella, diseñamos una heurística constructiva. La misma va asignando viajes a J-jornadas de manera golosa, hasta alcanzar una partición de V en J-jornadas.

Inicialmente los viajes se ordenan de manera creciente en cantidad de colores de sus respectivas listas. Para cada viaje no asignado (nos referiremos a ellos como *libres*) se intenta construir una J-jornada que lo incluya, es decir un camino en el digrafo D con inicio-fin en J y que pase por dicho viaje. Cuando no sea posible construir más jornadas y aún queden viajes libres, se realiza lo siguiente. Se marcan los viajes de tres J-jornadas de la partición como libres y se intenta construir J-jornadas distintas de manera tal de cubrir una mayor cantidad de viajes.

Iterando de esta forma conseguimos una solución con 214 jornadas, todas ellas legales y con 5990 minutos extras. El tiempo de ejecución fue de 304 segundos. Aunque esta no es una solución muy útil para la empresa (ya que posee más conductores y más minutos extras) al menos demuestra la existencia de una solución con todas jornadas legales.

Ahora consideremos una heurística de dos fases, cuya primera consiste en ejecutar la heurística constructiva, y la segunda en aplicar la heurística de mejora a la solución obtenida. Observe que esta nueva heurística siempre generará soluciones sin jornadas fuera de convenio.

Con esta nueva heurística se obtuvo una partición en 194 jornadas y 2788 minutos extras. Con el objetivo de reducir la cantidad de jornadas, en la segunda fase se aplicó los valores $n = 8, 6, 4, 3, 2$, es decir, generando subinstancias de hasta 400 viajes, motivo por el cual se requirió un tiempo mayor durante las optimizaciones. Su tiempo total de ejecución fue de 30 horas.

Si bien la cantidad de minutos extras disminuye significativamente respecto a la mejor solución encontrada hasta el momento (3245 minutos extras y 191 jornadas) requiere 3 conductores adicionales y esto no es admitido por la empresa.

Surge la pregunta de si existirá una solución sin jornadas fuera de convenio que requiera a lo sumo 191 conductores. Con la finalidad de que se utilice la menor cantidad de jornadas posibles en cada subinstancia evaluada durante la segunda fase, decidimos aumentar al doble el costo fijo. Obtuvimos así una solución con 192 jornadas y 3523 minutos extras.

Claramente, la disminución de jornadas fuera de convenio es un objetivo que compromete el de minimización de horas extras. Este compromiso sólo puede ser resuelto con una cuantificación por parte de la empresa (no considerada hasta

el momento) respecto al *costo* que implica utilizar jornadas que no respeten las restricciones.

Independientemente de este compromiso, nos propusimos intentar construir con la heurística una solución con 191 jornadas legales, triplicando el costo fijo. Sin embargo, observamos que el problema se vuelve más difícil ya que el aumento en el costo fijo conlleva a un aumento considerable del tiempo requerido por la optimización de las subinstancias, resultando que en el término de 48 horas sólo se había logrado disminuir a 194 la cantidad de jornadas.

4 Conclusiones y Trabajos futuros

En este trabajo se presenta una nueva forma de modelar el DJLC que mostró tener mejor rendimiento que las formulaciones de PLE usadas en [6] y [7], permitiendo resolver instancias de mayor tamaño. Además, al usarse este nuevo modelo para obtener las soluciones óptimas de las subinstancias que se resuelven en una nueva heurística de mejora, se logran soluciones mejores a las obtenidas hasta el momento para la instancia con mayor cantidad de viajes de la empresa de transporte interurbano con la cual se inició el estudio de este problema.

Como trabajo futuro, nos proponemos aumentar el límite en el tamaño de instancias que son posibles resolver por optimalidad con M_3. En este sentido, planeamos el uso de rutinas de separación de desigualdades válidas de coloreo por listas (a partir de estudios poliedrales recientes [12]) y una estrategia de selección de variables que ya ha sido exitosa en el marco de algoritmos branch-and-cut para otros problemas de coloreo [11, 13].

Respecto a las instancias de la empresa, nos queda abierta la pregunta de si para la instancia estudiada en este trabajo existirá una solución sin jornadas fuera de convenio y que requiera a lo sumo 191 conductores. También estudiar mejores soluciones para las otras instancias de la empresa sobre las cuales suponemos que podremos mejorar significativamente las soluciones considerando que la analizada en este trabajo es la de mayor cantidad de viajes.

Agradecimientos. Este trabajo es financiado parcialmente por los subsidios PICT-2013-0586 (ANPCyT), PIP 0277 (CONICET) y PID ING538 (UNR).

References

1. R. Borndörfer, M Grötschel and U. Jäger, *Planning Problems in Public Transport*, Production Factor Mathematics, Part 2, 2010.
2. S. Weider, *Integration of Vehicle and Duty Scheduling in Public Transport*, PhD thesis, Technische Universität Berlin, 2007.
3. A. Ernst, H. Jiang, M. Krishnamoorthy, B. Owens and D. Sier, *Annotated Bibliography of Personnel Scheduling and Rostering*, Annals of Operations Research 127, 21-144, 2004.
4. M. Grötschel, R. Borndörfer, A. Löbel, *Duty Scheduling in Public Transit*, Mathematics - Key Technology for the Future, 653-674, 2003.

5. IVU Traffic Technologies AG. Retrieved from: http://www.ivu.de
6. M. E. Alvarado, G. Nasini and D. Severin, *Algoritmos de mejora para el Problema de Asignación Conductores-Viajes en una empresa de transporte*, X Congreso del Instituto Chileno de Investigación Operativa ICHIO (OPTIMA). Concepción, Chile, 2013.
7. G. Nasini and D. Severin, *Particionamiento de Nodos de un Digrafo en Circuitos: Modelos y Aplicación a la Asignación Conductores-Viajes en una Empresa de Transporte*, XLVII Simpósio Brasileiro de Pesquisa Operacional (SBPO). Porto de Galinhas, Brasil, 2015.
8. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, W.H. Freeman, 1979.
9. IBM ILOG CPLEX Optimization Studio. Retrieved from: https://www.ibm.com/us-en/marketplace/ibm-ilog-cplex
10. Z. Tuza, *Graph colorings with local constraints - a survey*, Discussiones Mathematicae Graph Theory **17** (1997) 161–228.
11. I. Méndez-Díaz and P. Zabala, *A branch-and-cut algorithm for graph coloring*, Discrete Applied Mathematics **154** (2006) 826–847.
12. D. Delle Donne, *Estudios poliedrales de problemas de coloreo de grafos*, Tesis doctoral, Universidad de Buenos Aires, Argentina, 2016.
13. D. Severin, *Estudio poliedral y algoritmo branch-and-cut para el problema de coloreo equitativo en grafos*, Tesis doctoral, Universidad de Buenos Aires, Argentina, 2012.

	M_2		M_3	
Viajes	Nodos o Gap	Tiempo (seg.)	Nodos o Gap	Tiempo (seg.)
44	12	0,1	0	0,1
52	224	0,5	0	0,1
45	10	0,1	0	0,1
62	1334	4,8	0	0,1
52	1249	3,1	0	0,1
45	914	1,6	0	0,1
51	196	0,7	0	0,1
46	16	0,1	0	0,1
45	7	0,1	0	0,1
47	80	0,3	0	0,1
76	26515	90	0	0,1
78	71532	389	0	0,1
72	22010	75	0	0,1
68	16292	36	0	0,2
76	21805	55	0	0,1
75	64788	374	0	0,1
82	1895	10	0	0,1
71	5732	19	0	0,1
66	1799	4	0	0,1
71	80342	224	9	0,2
104	–	*7200*	3	1,6
99	87783	871	0	0,5
85	3,60%	*7200*	0	0,1
100	–	*7200*	0	0,2
107	–	*7200*	0	0,4
99	–	*7200*	2	2,3
93	–	*7200*	14	2,7
92	2,96%	*7200*	4	0,5
95	–	*7200*	0	0,2
88	0,73%	*7200*	0	0,2

Table 1. Comparación M_2 vs. M_3

	M_3	
Viajes	Nodos o Gap	Tiempo (seg.)
229	166	504
237	320	461
235	2380	1819
229	1399	1066
233	50	59
234	227	266
235	0	50
238	96	392
231	62	79
224	0	23
259	374	887
284	6130	3960
275	4633	4581
279	221	1076
282	0,14%	*7200*
272	1000	1179
289	4158	5142
285	0	191
272	897	1863
285	6807	7097
324	0,43%	*7200*
320	0,30%	*7200*
322	0,80%	*7200*
330	2005	5585
331	0,62%	*7200*
322	0,79%	*7200*
324	1042	3336
319	0,45%	*7200*
333	0,71%	*7200*
315	0,32%	*7200*

Table 2. Límite de M_3